Haaga-Helia
University of Applied Sciences

Modern web development with JAMstack

Edit Orosz

Bachelor's Thesis
Degree Program in Business
Information Technology

14-05-2020

Abstract

Haaga-Helia

Date 14-May-2020

Author(s)

Edit Orosz

Degree Program Business Information Technology

Report/thesis title	**Number of pages and appendix pages**
Modern Web Development with JAMstack	37 + 11

The reason for carrying out this research is the fact that it is beneficial for a software developer to be up to date about new technologies. JAMstack is a relatively new term in web development, that appeared around 2015 and there has not been widely researched in English which can be found in either Theseus or Haaga-Helia's internal thesis database. The aim of the research is to confirm the benefits of JAMstack written in the relevant literature and to bring up new aspects about what are the advantages of using it and are there any possible disadvantages of choosing this architecture. The research is also collecting data for whom is it beneficial to use JAMstack. The first part of the thesis is containing an introduction to the research topic, the methods and the basic definitions. The second part is the theory already available in books and online sources. It is discussing the basic building blocks, features, advantages, and usage of the architecture.

The third part is the research with is collecting data from primary sources, both quantitative and qualitative. There has been a survey and several interviews carried out to gather the data from experts in the software development field.

The collected data is gathered and analysed according to the main research questions. They confirm the already existing data and contain additional information that can be implemented by web developers and companies in future decision making either to choose JAMstack architecture or something else is more suitable for their projects.

The conclusion is that the four main benefits of JAMstack stated in the existing literature are valid and there are multiple more benefits experienced by the developers. The size of the project will distinguish if JAMstack is suitable for it or not. JAMstack can be beneficial for developers, companies, and clients as well.

Keywords

JavaScript, API, Markup, Web Development, Serverless, Static, CMS

Table of contents

Abbreviations

JAM	JavaScript, API, Markup
CSS	Cascading Style Sheet
HTML	Hypertext Markup Language
CMS	Content Management System
API	Application Programming Interface
FTP	File Transfer Protocol
LAMP	Linux Apache MySQL PHP architecture
MAMP	Mac OS Apache MySQL PHP architecture
WAMP	Windows Apache MySQL PHP architecture
PHP	Personal Home Page tools
MySQL	"My", the name of co-founder Michael Widenius's daughter, "SQL": Structured Query Language.
JS	JavaScript
CDN	Content Delivery Networks
MEAN	MongoDB, Express.js, AngularJS, Node.js architecture
SEO	Search Engine Optimization
PWA	Progressive Web Application
AWS	Amazon Web Services
NPM	Node Package Manager
SSG	Static Site Generator

1 Introduction

Technology is improving and changing rapidly. For a software developer or a tech-savvy person who must catch up with the latest tools, it is essential to find the most efficient solutions available. To understand the present, we must know where technologies are

coming from and what are the main elements, benefits, and disadvantages of choosing them. “…everything has a past. Everything – a person, an object, a word, everything. If you do not know the past, you can’t understand the present and plan properly for the future.” (Potok, 1996.) To understand the size of the difference in web development architecture between dynamical and JAMstack methods and how revolutionary, yet, not quite a brand new way JAMstack is re-introducing, we need to remember the past of web applications, how they had begun.

First, there were only quickly rendered and easy to build HTML sites, later CSS gave some style and life to the websites. From the late '90s, sites were running on Linux servers and using MySQL databases, the backend code was written in PHP. These are called LAMP and the versions which were running on Windows or macOS servers are the WAMP or MAMP. Since they were used, the term static site was not widely used anymore, since LAMP (and the others) were providing dynamical pages.
All this trend stayed until 2016. That was the year when the term JAMstack first appeared and from 2017, slowly but with a growing trend, it makes static sites “great again” (Dąbrowski, 2019.).

JAMstack stands for JavaScript, API, and Markup. JavaScript is one of the most popular programming languages. It is often used not only as a vanilla JS but also with frameworks and libraries, such as React, Vue, Angular. It is a client-based request and response dynamic.

APIs can be from a third-party application to a separately developed backend, that is deployed somewhere, or a simple backend created for the application, which is re-useable. The main point of APIs that they do not contain a database and do not require a backend engine on the server. They render static sites.

Markup language has a relatively short learning curve. It is ideal for content creators or anyone who does not know any other programming languages, someone who is not a technical person, such as a content-creator of a site, like a journalist, blogger, and so on. It is a static site generator and besides the markup, there can be its CSS style created with it or HTML elements. Nowadays, multiple frameworks help to create the template, like Gatsby, Jekyll, Hugo (Dąbrowski, 2019.).

What is JAMstack exactly? In brief, they are built usually with a static site generator, such as the above-mentioned frameworks and they do not depend on any web server. To

understand the whole picture, we must now also what JAMstack is not, since one can be easily confused. "A site built with a server-side CMS like WordPress, Drupal, Joomla, or Squarespace. A single-page app that uses isomorphic rendering to build views on the server at runtime. A monolithic server-run web app that relies on Ruby, Node, or another backend language."-as jamstack.org, the official site for the architecture puts together.

The thesis attempts to find answers to the main research questions about JAMstack, which are the following:

1. What are the benefits of developing a modern web application according to JAMstack which makes it stand out and simplifies web development?

2. What are the pros and cons of developing applications with JAMstack compared to other methodologies?

3. Who benefits from building an application according to JAMstack architecture?

As jamstack.org and many other sources state, using JAMstack has four main benefits:

1. Cheaper, easier scaling: CDN's scaling is much easier compared to scaling traditional web applications since the files can be served anywhere.
2. Better performance: Generating a site and deployment is faster than traditional methodologies. Being served on a CDN is making the difference in a great performance.
3. Higher security: If there is less possibility to attack from the outside, there is higher security. That is made possible with the server-side rendering.
4. Better user and developer experience: The development and debugging are made easier because the front-and back-end is decoupled. Through the CMS, there is no need for an extra content stack, and marketing. There is a possibility for offline apps and native-like user interfaces (jamstack.org, 2020.).
   These statements are proved or disproved by evidence through written resources, interviews, and a survey.

2 Theoretical background

What is JAMstak? It can be examined in two ways, to begin with. What does it stand for and what does it mean?

The word is coming from JavaScript API and Markup. The word stack means the layer of technology in which way a website or application is delivered. JavaScript, according to the definition of developer.mozzilla.com: "JavaScript (JS) is a lightweight, interpreted, or just-in-time compiled programming language with first-class functions. While it is most well-known as the scripting language for web pages, many nonbrowser environments also use it, such as Node.js, Apache CouchDB, and Adobe Acrobat. JavaScript is a prototype-based, multi-paradigm, single-threaded, dynamic language, supporting object-oriented, imperative, and declarative (e.g. functional programming) styles." (developer.mozzilla.com, 2020). API is controlling how the component of the application or web site or software is communicating with each other. Markup is a language that is easy to learn for non-programmers as well. It contains tags to build up the content structure. Probably the most well-known are the XML, HTML and there are others, such as BBC and SGML.

When the actual use of JAMstack is explored, the benefits of using it and the main features of it can be summed up such as “Fast and secure sites and apps delivered by pre-rendering files and serving them directly from a CDN, removing the requirement to manage or run web servers.”(jamstack.org, 2020.).

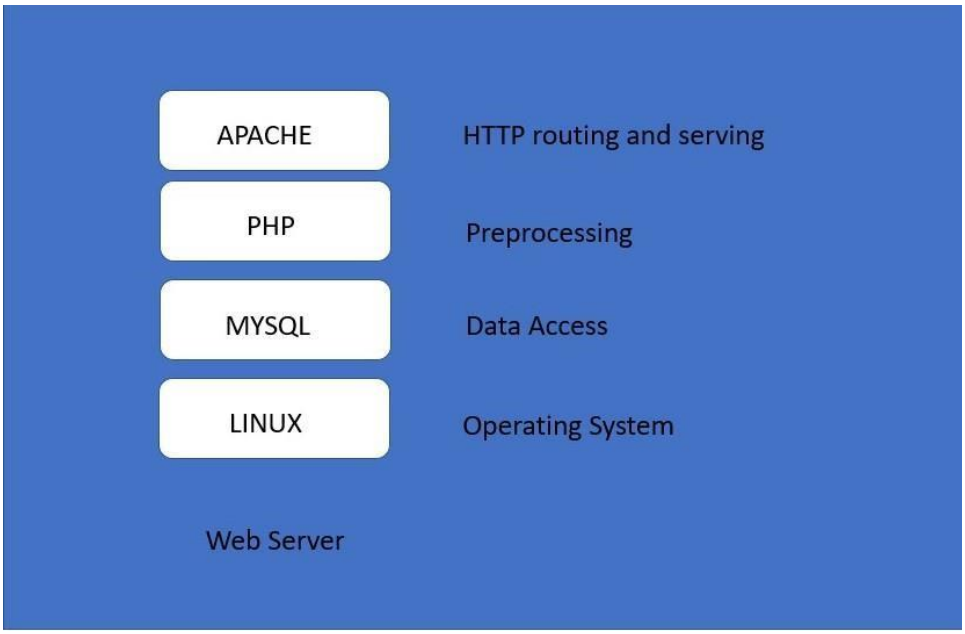

Figure 1 LAMP architecture (Hawksworth, 2020.)

In the early 2000s has begun the popularity of a traditional LAMP architecture (Figure 1), everything is at least on one web server, but it needs most commonly multiple web servers as the load increases. Sometimes they can abstract the data layer for example on a separate server. It's down to the operating system through routing and the actual serving.

The other - since 2013 - popular architecture is MEAN (MongoDB, Express.js, AngularJS, and Node.js) which is working similarly as explained below about the LAMP. However, MEAN is more supportive with a responsive UI, NoSQL databases, and communicates with one language (JavaScript) both on back-and front-end (Myers, 2018.).

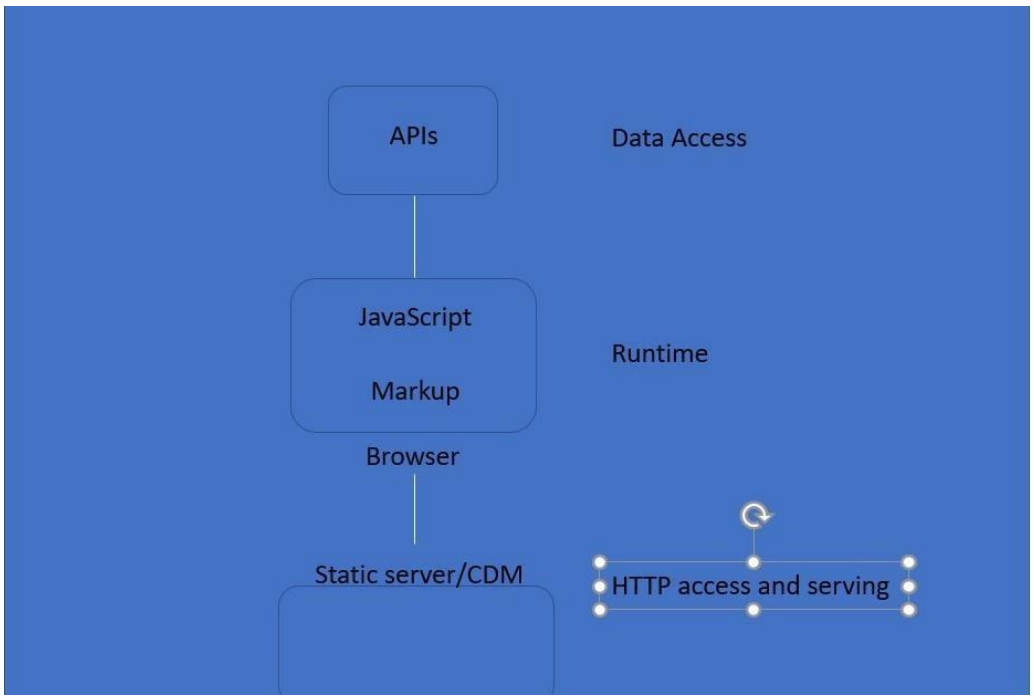

Figure 2 JAMstack architecture (Hawksworth, 2020.)

In a JAMstack architecture, there is still HTTP serving, there is still a need for accessing data and still need to process in a runtime. The main difference is that the "operating system", the processing runtime is working through the browser. There is no need to go down to an actual operatic system, like in LAMP, but it can operate in the browser. The API isn't the part of the domain in this case. There is no need to manage the API and the data coming from the API, it can work on a separate system, only a request needs to be sent to fetch the data from the API, which is happening through JavaScript functions. The

serving is pre-generated so it can be a very simple server or a CDN which can be abstracted away since there is no internal logic in the static server.
The API, JS, and Markup are taking all the necessary responsibilities and it is moved away from the server.

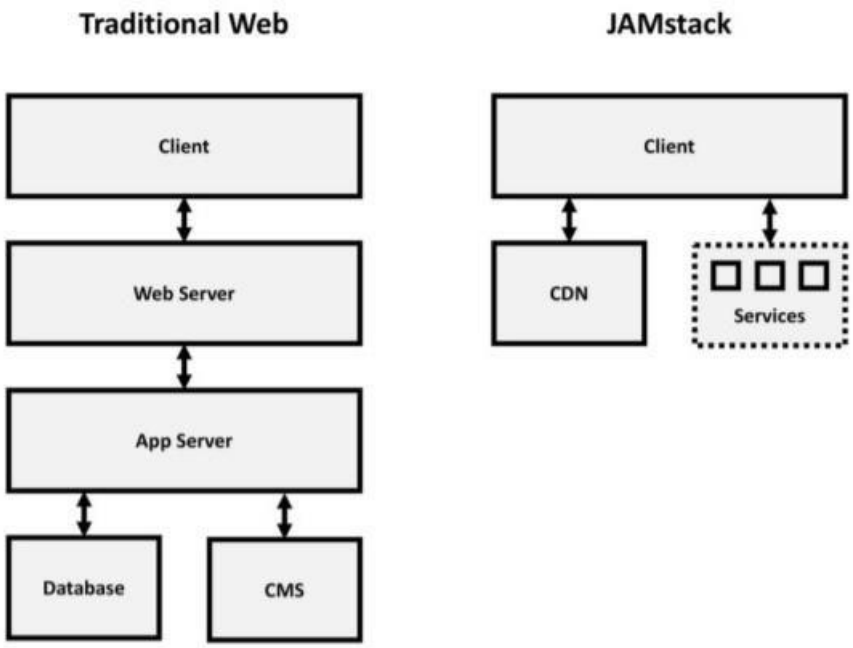

Figure 3 Traditional Architecture VS JAMstack (Bull, M. 2017.)

Figure 3 shows the key differences between a traditional and JAMstack architecture, how the application or website is relying on a server, and how the complexion of servers reduced to a client + services simplicity. (Bull, M., 2017.)
There is another main difference in JAMstack, the whole development is in a Git repository. The config, the CSS, the, JS is all there, and everything is pushed to the repository, so all is under version control and the source is centralized. Instead of generating new content for each call, with JAMstack the HTML is already created, and a static file and it is sent to the client and it doesn't need further processing which makes rendering and request-handling faster. (Gienow, M. 2017.)

2.1 The elements of JAM

- JavaScript:

Brendan Eich created JavaScript in 1995. After constantly coding for 10 days in a row. He called it Mocha at first. Today, it is almost exclusively used by internet browsers. JS has all that one can need, from classes, objects, destructive functions, modules, and so on

and also includes the nowadays most-used data format on the internet, JSON. (Biilman & Hawksworth, 2020.,8.)

- API:

Roy Fielding wrote a dissertation in 2000. which can be the origin of APIs: “Architectural Styles and the Design of Network-Based Software Architectures.” (Biilman & Hawksworth, 2020., 9.). “…defined Representational State Transfer (REST) as a scalable and discoverable architecture for services exposed through HTTP.” (Biilman & Hawksworth, 2020., 9.) The first API needed either a proxy or a Flash or Java Applet to communicate with another API outside of the browser. JavaScript became a complete runtime environment and new standards became widely used which made APIs accessible through the browser when using a JavaScript client. (Biilman & Hawksworth, 2020., 9.)

- Markup:

At the beginning of web development, there were only HTML files running on the server which run through the HTTP protocol. Later, due to the development of the web and therefore, the change of requirements, it has become a program ran on web servers which built an HTML site connecting to the database simultaneously, on the fly for each request. However, it was significantly slower than simply rendering static sites, but this was the only way due to the browser’s ability, only to view the documents. When there was any kind of interaction from the user, such as writing a comment, clicking on a submit button, or making any interaction with the site which would result in a change of view, the server needed to create a new HTML document and render the whole layout again. It was not a responsive web application, as we are used to using these days.

JAMstack is pre-building the markup and serving it directly to the browser from a CDN. The process requires a static site generator like Gatsby.js, Next.js, or Jekyll or building tool like one of the most widely used, WebPack. This way the files written in markup language are transferred to be an HTML file and the source code is JavaScript and CSS. (Biilman & Hawksworth, 2020., 11.) This approach makes it easy to implement an efficient SEO or to create progressive web applications (PWA) which makes possible to install the whole front-end application on any kind of device.

Because the front-end service is becoming more and more independent from the back end and can use multiple APIs or multiple front-end applications can re-use the same API, this makes it possible to create more efficient microservices for the back end. This is one reason, nowadays trending cloud micro-services are existing, such as Lambda and others from AWS. (Biilman & Hawksworth, 2020., 12.)

2.2 Types of JAMstack projects

Using JAMstack architecture can be beneficial for many kinds of projects. For example, they can be version-controlled static sites. The HTML files and JavaScript, CSS, and other assents are all pushed to Git. If there are more elements, like navigation, components used, they are in their container component. If there are more modules needed, they can be simply added through NPM. Everything is deployed continuously through Git.

The content-editing can also happen through CMS, content-management platforms. It can be two different approaches, either through git commands, like Netlifly, Forestry, or prose or it can be an outsider-API, such as Contentful or Prismic.

The usage of JAMstack at all sizes of web applications is making publishing significantly faster. The early SPA's became more widely used after JavaScript was evolved enough for better performance. The first SPA's were a part of a monolith-application and serverdependent. Nowadays, the front-and back-end is getting more separate, the front-end applications are stand-alone and can be working together with different APIs. (Biilman & Hawksworth, 2020., 13.)

The build-tools are much faster and static site generators are rendering thousands of pages under a second and the whole building cycle goes faster. Content creators can publish an update much faster and therefore be much more competent with each other and provide up-to-date content that can be necessary relating to certain services such as governmental, news, health, or weather portals.

For larger companies, which need to create multiple types of content, it has been challenging to maintain the same look over their website, blog, message board earlier when the front-end was coupled with the back end. With JAMstack, the same front-end can be built for all applications and they can work with separate APIs so the look and feel stays the same, loyal to the company’s branding, and the functionality is always applied separately to certain content. (Biilman & Hawksworth, 2020., 15.)

2.3 The advantages of JAMstack

There are multiple reasons why companies are moving towards JAMstack architecture instead of traditional methods. This makes it possible for smaller teams to work on a complete project which can be completed on a shorter time due to less complexity is

needed in the architecture, no need to build a back-end infrastructure, and care about servers.

The scalability is cheaper because of these reasons. The cooperation with multiple vendors can go easier because they can share or re-use APIs. Because of all of this, releasing any product can be quicker, the development is more agile, and updates are more continuous and easier also from a third-party when it comes to content updates. (Carlise, M., 2018.)

- Cheaper and easier scaling:

  Because of the less complex structure, even a smaller agile full-stack team can work on a JAMstack product, use only JS and Markup, and deliver it in a shorter time. As the applications are getting more complex, more and more skills are needed for the front-, and back-end specialists, that is why decoupling the two sides can be a much more efficient approach. The API services are more re-usable and easier to scale and use for different services. Much fewer resources needed if the architecture is so much simpler. This makes the costs all-together lower.

-

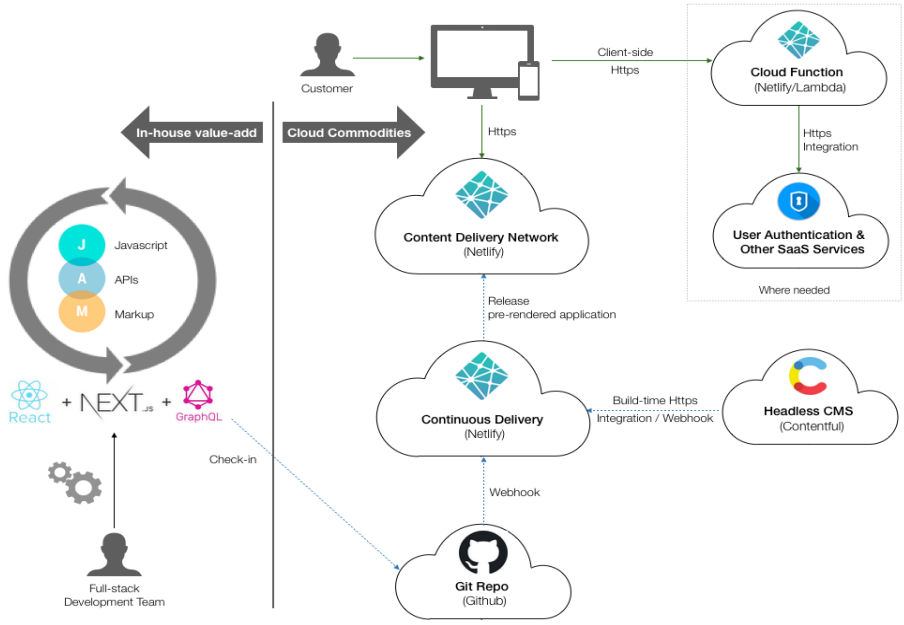

Figure 4 Illustrating how the benefits of JAMstack can happen through its infrastructure (Carlise, M., 2018.)

- Better performance:

  When there is a less complex architecture, the development is faster, so as the up-todate deployment and only the Markup needs to be updated in that case. When less code must be run, it will require less time so all the updates are much faster and the

site loads will be faster as well, especially, because it is a static site. There is no need for database queries which means also faster loading (Biilman & Hawksworth, 2020., 19.).

- Higher security:

  “When considering security, the term surface area is often used to describe the amount of code, of infrastructure, and the scale of the logical architecture at play in a system.” (Biilman & Hawksworth, 2020., 29.) As had been mentioned above, the architecture model is much simpler with JAMstack. Compared to a LAMP or other architecture that is more complex, there are fewer possibilities to fail, fewer segments to collapse. The maintenance and ability to see-through are better with a simpler architecture like JAMstack.

Figure 5 Demonstrating the reason for better security with JAMstack (Dionne, 2020.)

The back-end APIs are also read-only which makes them more secure. There is no logic needed to be executed which is beneficial for both time- and safety concerns. To demonstrate it, JAMstack can be compared to a WordPress project. “WordPress sites as a useful comparison. A WordPress site combines a database, layers of logic written in PHP, templates for presentation, and a user interface for configuring, populating, and managing the site. This user interface is accessed over the web via HTTP, requiring that a WordPress site is capable of accepting HTTP POST requests, and consuming and parsing the data submitted to it in those requests” (Biilman & Hawksworth, 2020., 30.) JAMstack doesn’t have many layers to guard and the back-end is only readable, with no database queries to handle. A huge part of a JAMstack service can be outsourced and secured in additional services. A big part of the whole project’s code is read-only. Because of the separate APIs, most services are not accessible for the public, only for the build-environment. (Biilman & Hawksworth, 2020., 31.)

- Better user and developer experience:

  Because of the separated back- and front end, not only developers can focus on one part at a time and implement better microservices but also debugging becomes easier. If someone has ever coded, they must have noticed, even in a smaller amount of code it can be tricky to find what causes the error. Debugging a larger project can

take sometimes more time than writing it in the beginning. That is why less amount of code and more decoupled parts are providing a better ground to solve errors.

Figure 6 The logical chain to better user experience with JAMstack (Dionne, 2020.)

The lesser code which needs to be executed at a user request and the nature of the static site makes the site faster. It is resulting in better user experience and that is beneficial for the company, as they will get better search results and therefore more traffic for their service or application.

3 Research methods

3.1 Collecting primary data

For the research, there have been different methods of data-gathering carried out. The main goal was to collect as much primary data as possible. There are two main types of data sources according to Walliman (69., 2011.), primary and secondary data sources. Primary data is the most accurate therefore valuable for research. They are experienced or measured data first-hand. Secondary data is what has already been measured and the secondary source is referring to it. Therefore, secondary data is less reliable because the collected information can be interpreted in different ways. Because of this, this research is using primary data.

The other main categorization of data (Walliman 71., 2011.) is either quantitative or qualitative. The first one, quantitative means that the results can be written as numbers, they are measurable according to mathematics, average, median calculation, etc. Qualitative data refers to something which can be described with words, expressions, feelings, etc. Those findings cannot be measured through mathematical formulas, they need different ways to conclude from (Walliman, N., 71., 2011.). Because of these reasons, both types of data are used. They both show different aspects of the research topic. This research is implementing a survey and interviews to gather primary data. The survey is gathering both quantitative and qualitative data, the interviews are qualitative data sources.

The research methods had been chosen to attempt a valid display of the current situation of the usage of JAMstack. The first method is a questionnaire with 16 questions which covers the answerers' background, experience, and certain aspects related to JAMstack. The questions are created to be able to understand only by professionals in web development or those who are studying about it. It had been shared on multiple social platforms to reach to as many people as possible and gather information from different backgrounds, levels of experience, or geographic location. This way the data have been collected from a wide range of software developers and students and show an up-to-date picture of the usage of JAMstack and views on the topic.

The other research method chosen is an interview which had been carried out online. This way there could be credible data collected from known individual IT professionals, altogether from five software developers with a strong background. Most of the

interviewees have from 10 to 20 and more years of work experience. This way, the data comes from
the most credible sources. The interviews are also a good example to show the experience of JAMstack, prove the necessity, or in one case, represent the opposite opinion. One source is not a source, that is why the opposite must be as well taken into consideration. The interviews and the survey give freedom of expression and both sides are represented.

3.2 The survey

The survey (URL: https://forms.gle/GqDZJEBDjRfKYyBF9) has been made with Google Forms, because it is free to use, provides downloadable data visualizations and unlimited answers can be submitted. The researcher had shared the survey on multiple platforms online, such as LinkedIn, Facebook, Instagram, JAMstack forums, and friends, to reach out to as many software developers and information technology students as possible.

For the quantitative data of the survey, there are multiple graphs are available through Google Forms to visualize and measure the gathered data. The qualitative data can be interpreted through logical explanation, according to the meaning and what they are referring to.

3.3 The interviews

Due to the restrictions by WHO and the Finnish Government of personal contact during the 2020 coronavirus pandemic, which does not allow people to meet in person, the interview took place through e-mails. There have been five senior developers interviewed. The answers have been gathered through March 2020.The full interviews can be found attached in the Appendix. All the developers, except one, included their names, years of experience, and the companies where they are working at currently.

4 Results of the research

There were 26 developers and students who completed the survey. The questions had been designed to give a general overview of the respondents and to give answers to the three main research questions. The questions of the interviews and the survey are grouped according to the information they can provide. In the first group, some questions

help to have a better view of the respondents' background and experience. The other three groups containing those answers, which can answer the research questions.

4.1 Overview of the respondents

The first 6 questions of the survey are regarding the answerer's background. The questions intended to give a picture of those who answer and show if their background and experience can be relevant enough to give valuable information about JAMstack. The interviews were made with known developers, that is why only their years of experience and knowledge about different stacks were asked regarding their background and to validate the information's value that they were providing.

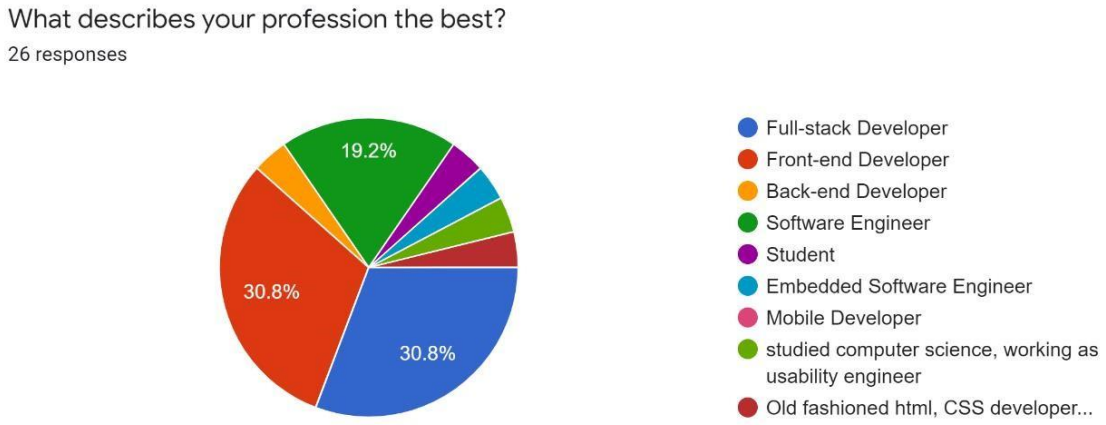

Figure 7 1st question of the survey

The first question is describing the answerers' background. Most of the answerer's profession, all-together, 61,6%, of which 30,8% are Full-Stack developers and 30,8% are FrontEnd developers which are the best target group to ask about JAMstack. Each of the student, back-end developer, software engineer, embedded software engineer, usability engineer, "old fashioned HTML, CSS developer", as added as another answer option, has 3,8% amongst the answers.

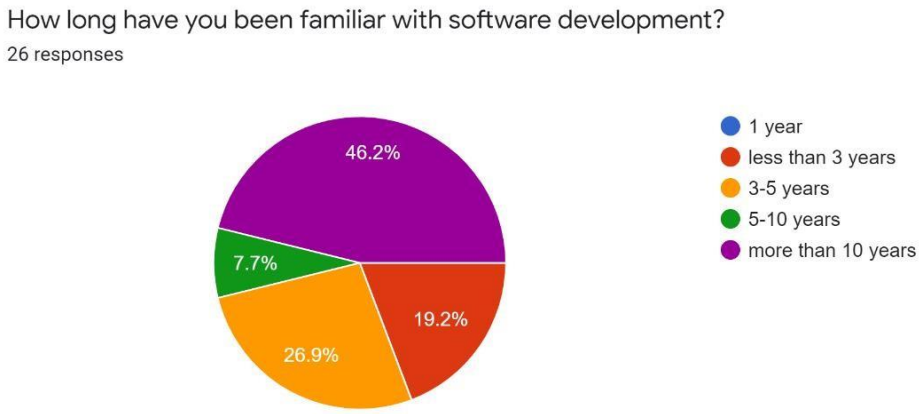

Figure 8 2nd question of the survey

The question is referring to the years of experience the developers have. The biggest number of respondents, 46,2%, have more than 10 years of work experience, the second largest group of answerers 26,9% has 3-5 years of work experience. 19,2% of the respondents have less than 3 years of experience, 7,7% have 5-10 years of experience. The result is showing that most of the respondents have long years of experience in software development.

Work experience of the interviewees:

Name	Experience in years all / in web development
Sauli Purhonen	7
Andreas Koutsoukos	17 /13
H. Tuohimaa	14
Joe Smallwood	10/4
Anonymous	20 /4

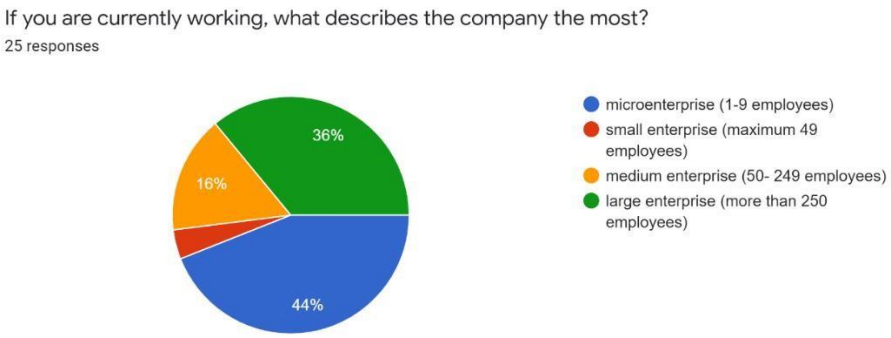

Figure 9 3rd question of the survey

The next question is related to the company-size where the developer is working, to highlight the usage of JAMstack at different sized companies. The most answers, 44% came from developers working at micro-enterprises and a significant amount, 36% are working at large enterprises. 16% of the answers are from people working at medium-sized enterprises and 4% are working at small enterprises.

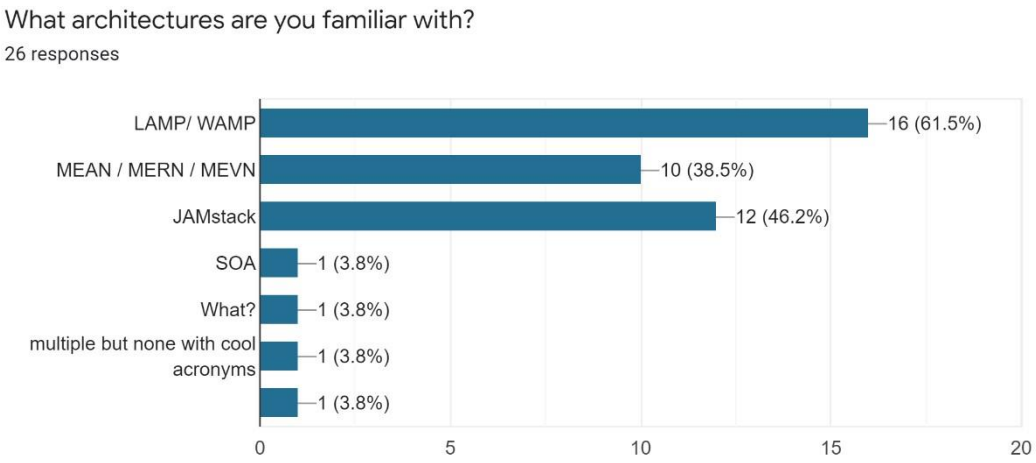

Figure 10 4th question if the survey

This question is asking about the background of the developers, what architectures have they used before, and how many of them are familiar with JAMstack. 16 people, 61,5% are familiar with LAMP/WAMP architectures, 38,5% of the people are familiar with MEAN/MERN/MEVN stack, 12 people, 46,2% are familiar with JAMstack as well. SOA and multiple others are known amongst 3,8% of the respondents.

The interviews also had two questions related to usage and knowledge about different stacks amongst the developers:

1. What was your first stack when you had begun working?
2. What stack are you using currently?

The full answers can be found in the appendices.

Name	Stacks
Sauli Purhonen	1: Lamp stack, WordPress
	2: Angular with Django API
Andreas Koutsoukos	1: LAMP stack
	2: JAMstack
H. Tuohimaa:	1: LAMP/WAMP stack
	2: isomorphic JS with Vue or React and Node.js backend, JAMstack
Joe Smallwood	1: MEAN stack
	2: JAMstack
Anonymous	1: WAMP
	2: LAMP

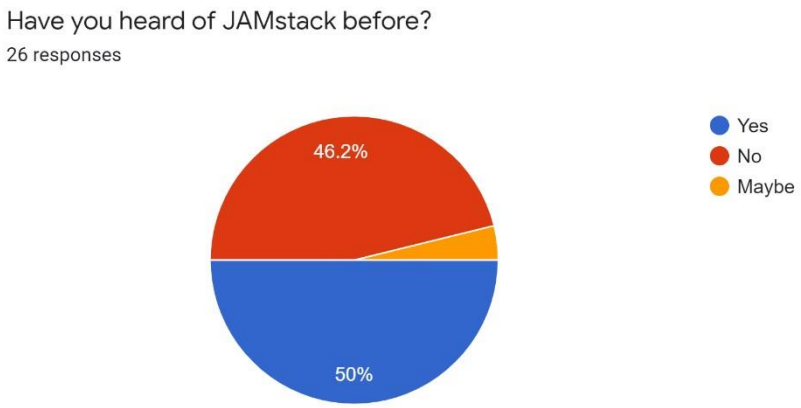

Figure 11 5th question of the survey

This question points out the previous knowledge about the existence of JAMstack. 50% of the developers have heard of JAMstack, 46,2% have not heard of it and 3,8% are not sure.

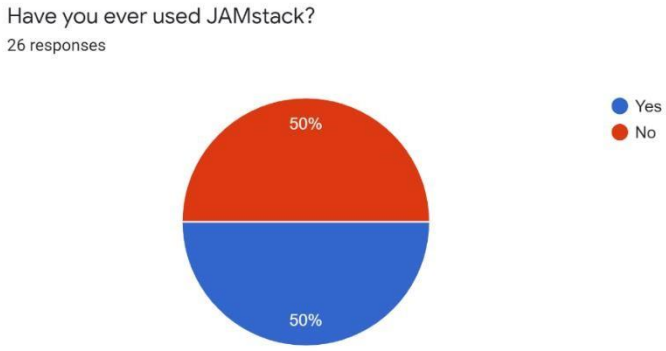

Figure 12 6th question of the survey

The 6th question is asking directly about the usage of JAMstack and 50% of the answerers have had used JAMstack at some point while half of the people who answered, have not used it yet.

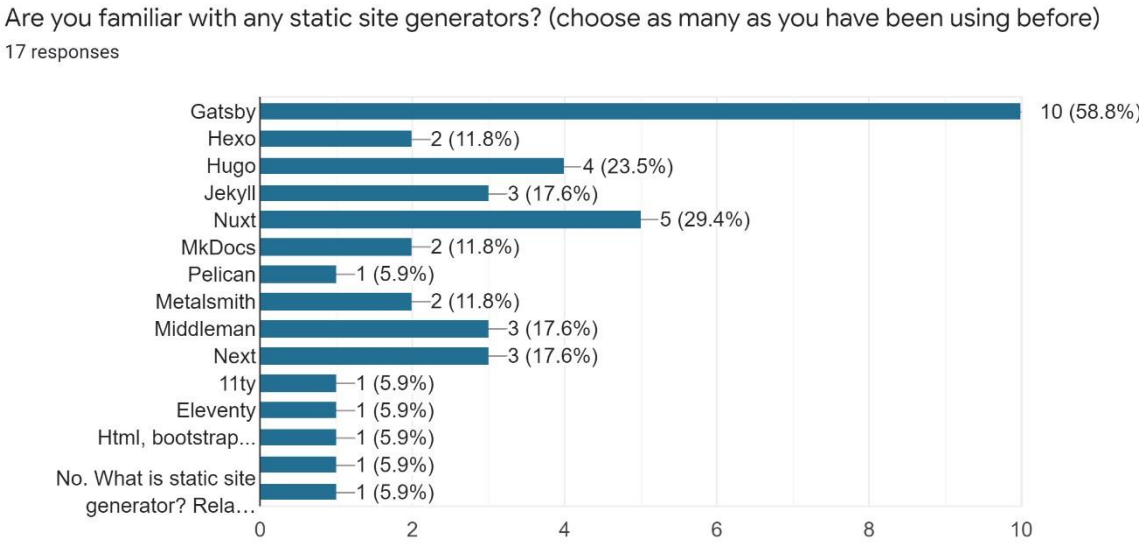

Figure 13 11th question of the survey

The question is asking about the familiarity with site generators. Most of the respondents have experience with Gatsby, 10 answers, 58,8% followed by Nuxt, with 4 answers, 29,4%, and Hugo on the 3rd place with 4 answers, 23,5%. 17,6% of the respondents chose Jekyll, 17,6% selected both Middleman and Next, 11,8% selected Hexo, MkDocs, Metalsmith and 11ty, Eleventy, got 5,9% each. Those are the same but named differently

by the answerers so that adds up to 11,8%. There are two other answers added which are irrelevant to the question.

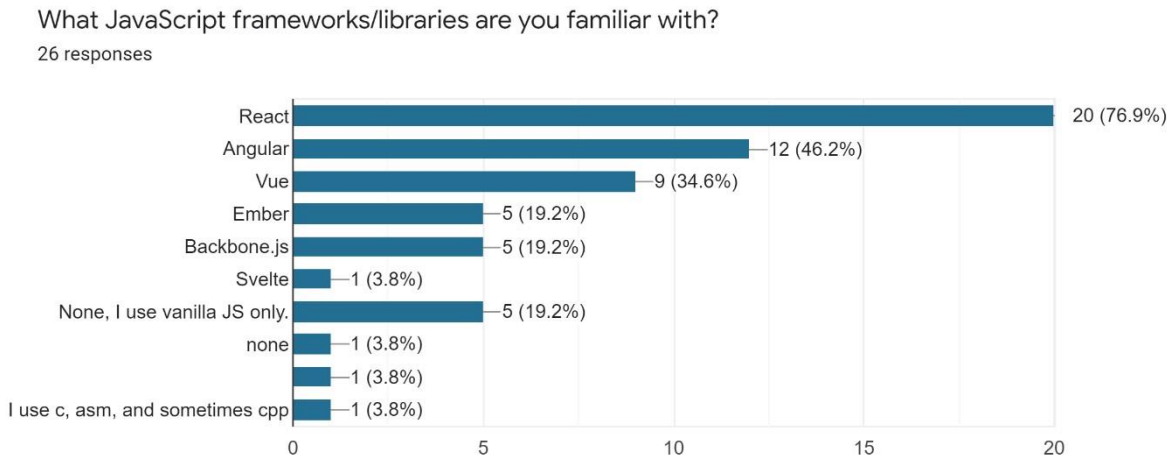

Figure 14 12th question of the survey

The answers to this question are showing a trend of JavaScript using, which libraries and frameworks are the most-used since one of the building-blocks of JAMstack is the script written in JS. According to the responses, React from Facebook is the most popular with 20 responses, 76,9%, right after Angular from Google with 12 responses, 46% and followed by the open-source Vue with 9 responses, 34,6%. Ember, Backbone.js, and only vanilla JavaScript are used each by 19,2% of the developers. Svelte and no frameworks at all got 3,8% and the other added answers are irrelevant.

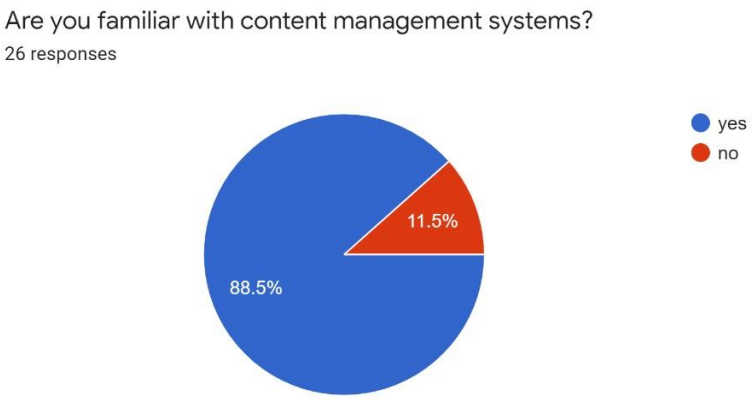

Figure 15 13th question of the survey

The question is asking about the familiarity of CMS since it is also one of the most common tools of using JAMstack. The majority (88,5%) of respondents is familiar with a CMS tool, 11,5% is not familiar with content management systems.

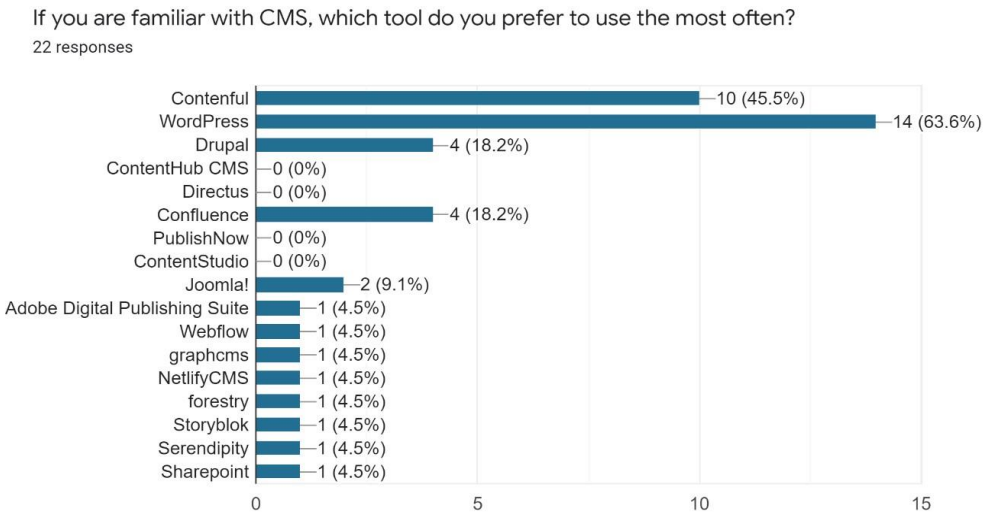

Figure 16 14th question of the survey

The question is asking directly about a preferred CMS tool and according to the answers, WordPress with 14 answers, 63,6%, followed by Contentful with 10 answers, 45,5%, are the most used ones. After those, Drupal and Confluence are both used amongst 18,2% of the answerers and Joomla! is used amongst 9,1% of the respondents. Other CMS systems were selected by 4,5% each, such as Adobe Digital Publishing Suite, Webflow, graphcmc, NetlifyCMS, forestry, Storyblok, Serendipity, Sharepoint.

4.2 Questions regarding the first research question

Both the survey and the interview contained questions regarding the first research question. What are the benefits of developing a modern web application according to JAMstack which makes it stand out and simplifies web development? The survey- and the interview- questions had been designed to find answers from different aspects of why it can be beneficial to use JAMstack. The survey and the interviews are addressing questions about the timeframe to learn it, the difficulties about getting familiar with it, the benefits of why a developer or a company would decide by using JAMstack.

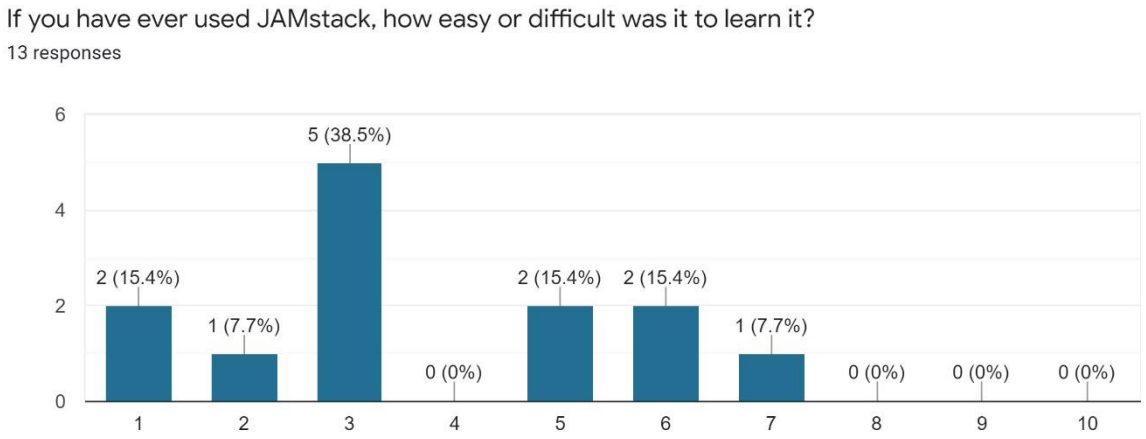

Figure 17 7th question of the survey

The question is about the learning-curve of JAMstack. There are references to JAMstack being easy to learn in multiple sources. According to the answers, it is rather easy to learn, 38,5% gave 3 out of 10 when 1 is very easy and 10 is very difficult. 15,4% for 1, so very easy to learn and 5 and 6 got 15,4% as well. 2 out of 10 got 7,7%, so as 7 out of 10.

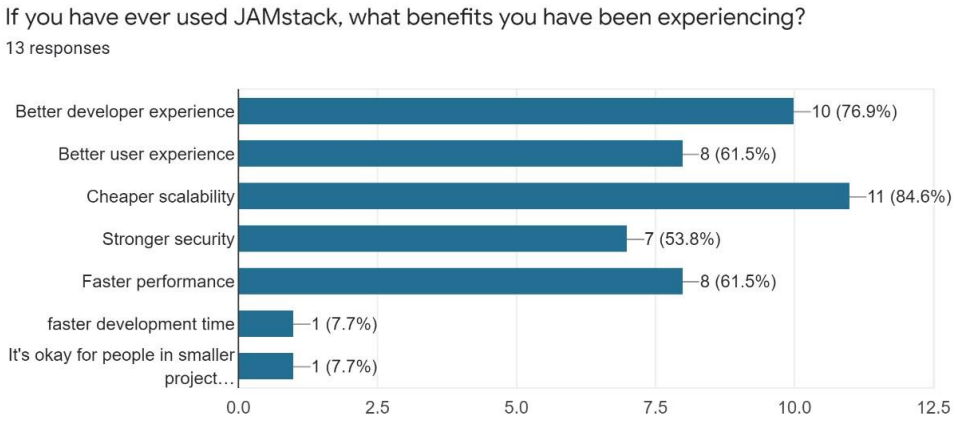

Figure 18 8th question of the survey

This question is asking about the commonly mentioned benefits of JAMstack or any additional one if they would like to mention more than listed. The most-commonly selected benefits are better developer experience, which are 10 answers, 76,9%, cheaper scalability with 11 answers, 84,6%. Faster performance and better user experience got both 8 answers, 61,5%. There are two additional benefits have been mentioned with both 7,7%.

The interview is also containing questions about the benefits of JAMstack:

1. Have you been familiar with JAMstack?

   If yes, what is your experience? How long did it take to learn it? Was it simpler or more difficult compared to other architecture (such as LAMP, WAMP, MEAN, etc.)? Can you explain the main advantages you have been noticing and possible disadvantages? What kinds of projects have you been using, or prefer to use it with?
2. Would you like to add anything?

The full and literal answers can be found in the appendices.

Name	Benefits
Sauli Purhonen	1: He has done a few projects with Netlify CMS, Strapi & Firebase. Prefers it because of the ease of use and integration with modern frameworks as Angular, Vue & React are made simple and could be used in many projects from light to medium use cases. For larger apps, he still going for the dedicated backend to keep all things under control.
Andreas Koutsoukos	1: He has been using JAMstack kind of architecture about 5 years ago when static architecture models were popular. It took about half a week to learn JAMstack. Compared to other stacks JAM is simple and good for professionals that are designers and code. Advantages are that sites are built-in less than a half-hour and then the code is pushed to GitHub repository and starts the build process in Netlify.

H. Tuohimaa	1: Yes, typical JAMstack with static site generation is easy to understand. After all, you need some frontend code and (markdown) content. This combo is pre-built to a static site and then deployed to CDN. Really easy steps to understand. Costs are cheap because there are no virtual machine needs. Sites are fast because of pre-rendered file architecture. When you must update a site, you can automate it with a small CI/CD pipeline.
	2: Usually these new stack architectures are much easier to understand than some old ones. They are also more straightforward to use and have a low learning curve.
Joe Smallwood	1: He has good experiences with Gatsby because of the ease of setting up a website quickly that is optimized out of the box for SEO and performance. There is a possibility to mix between creating a SPA with modern tools, but in the end, just generating static HTML files for different routes. The concept of JAMstack doesn't take much time to learn but Gatsby could have a steep learning curve for people unfamiliar with GraphQL or React. Way simpler and easier to set up than other architectures he had been working with. No time needs to be spent setting up complex servers and backend services. CI is much simpler since we are just generating static files that can be uploaded to a server or static hosting service like Netlify. With a good static hosting service, JAMstack sites are incredibly fast for the enduser. Serverless functions are also easy for frontend developers to understand since they are just JavaScript.

4.3 Questions regarding the second research question

The survey and the interviews are giving information that can be used to find answers to the second research question. What are the pros and cons of developing applications with JAMstack compared to other methodologies?

If you have used JAMstack before, have you experienced disadvantages compared to other architectures?
Text responses mentioning concrete disadvantages:

CDNs are somewhat reliable but I would rather have fewer dependencies in my application.
Harder to throw together complex full-stack applications
I come from a WordPress/PHP development background so now headless CMS's are where I experience knowledge gaps, but it does not take too much time to get up to speed with the options out there.

6 out of 10 respondents had been positive about not experiencing any disadvantages about using JAMstack.

What makes you decide which stack are you or your team are going to use for a project?
Text responses:

The simplest and adaptive is usually the best one. If you use JavaScript + NodeJS combo, you can do front- and backend without special knowledge
The project itself. The platform needs to be running on, the future supporting, scalability.
The project requirements and prior knowledge of technologies
I work on small sites, so right now I am defaulting to JAMstack to see how it goes.
Cheapness.
customer > usual usage in company > experience in team

Reliability on dependencies, easily implementable security, and learning curve among current and future employees as well as the ability to scale and reduce a possible spiderweb of coding.
Simplicity.
The knowledge of the team. The project budget and timeline.
Using JAMstack on personal projects as the business I work for does not want to change their stack. Going forward, I will use JAMstack for pretty much every project just because
Nicer developer experience.
Suitable to team members skills & easy to train others
Company policies, team’s competences, how easy it is to find new competent developers, development experience. Also, the project's requirements and lifetime expectance effect (for a small internal tool it’s safer to use some cool new tech instead of a well-established and battle-tested solution).

The answers are according to the pros and cons of JAMstack and telling about the pros of choosing JAMstack and how/why it is selected to use. 4 answers are mentioning that the company decides what to use and there are 4 answers which are mentioning that choosing JAMstack is also depends on the nature of the project. 5 answers are mentioning that it is nice to work with, it is easier for the developers, it provides a better experience to develop a project with. The project's budget, the cost-effectiveness, and simplicity of use and learning are also amongst the answers, as pros of JAMstack.
The interviews had questions to discover the pros and cos of JAMstack, as well.

1. (Have you used JAMstack before?) If no, what made you decide not to get familiar with/ use JAMstack architecture?

2. Are there any more observations, experiences, or advice related to JAMstack or comparing it to other architectures you have been using before?

The complete answers are in the Appendices.

Name	Answers
Andreas Koutsoukos	1: JAMstack is not the best for sites that make interaction from users' needs like Twitter, you can make Twitter in JAMstack but hard to say is it the best stack for that kind of site. 2: Time will tell and more I am doing JAMstack projects will tell me what the real benefits are.
H Tuohimaa	1: In a big project where the client is also updating content in a back service during development, JAMstack
	can start to be a little bit clumsy. Contentful + Netlify, if the content has to be updated in Contentful, you must build Netlify every time to see what you have done, and the build time can be multiple minutes. 2: JAMstack can work well in smaller sizes with a small team.
Joe Smallwood	1: For disadvantages, the main is not being able to easily live preview content changes from the CMS. Since we need to build the site when content changes there is always going to be some delay before seeing the changes live. Also, most SSGs require us to build the entire site every time content changes JAMstack does not make sense for very dynamic websites with content changing very frequently.
Anonymous	1: Specific platforms require specific tasks. Changing the current stack would need a big business decision. Decisions regarding changing platforms could mean too much risk.

4.4 Questions regarding the third research question

The third research question is seeking an answer to find who is JAMstack the most recommended to try out. Who benefits from building an application according to JAMstack

architecture? Is it good for both customers and developers? The questions in the interviews and the survey are attempting to give answers to this question as well.

If you have never wanted to familiarize yourself with JAMstack or never decided to use it, what were the main reasons for making that decision?

Text responses:

I never needed it yet.
Using CDNs more effectively.
JS-centric, large database projects require a different approach.
A programmer almost always bends things to their will. There is no absolute right or wrong decision about choosing the most flexible CMS or SSG is something that should be thought of well ahead.
Never heard of it. Not needed for embedded development.

According to the text responses, the most common reason was that they have never needed to use it, yet or that the projects required a different approach, because of a large database or using CDN in a different way or choosing the most flexible CMS or SSG.

Here you can add anything else freely which is related to the topic. You can tell about good/bad opinions and experiences or any advice for those who are learning web development or wish to find the best tools for creating the best possible results.

Text responses:

JAMstack is an easy, agile, simple, and adaptive way to do the job. It does not have to be always with static site generator (like the definition says), you can also do it with dynamic backend.
I guess, using premade UI-components can speed up a lot when creating a project from scratch.

Easy to learn and to deploy projects, a revolutionary approach. I suggest starting with the basics. Moving from creating your CRUD system to starting with easier systems like WordPress and then a CMS that can be more complex as it grows such as Contentful. It's also honestly what the person truly enjoys. It's always good to love what you're working with and not to dread ever touching it.
So, I started building websites back in early, the mid-90s, self-taught... have learned some stuff, trying to use some of the newfangled tools... one website I currently maintain will be redesigned with some sort of stack, but when I asked the developer how do I make changes (edits) to content daily, he couldn't answer me. So, not a fan right now ...
As we have been working with JAMstack principles since before Netlify re-branded static sites with that name, we have a lot of tooling that lets us work with them more easily. We have also produced a SaaS e-commerce system for JAMstack sites. So, we are not at all limited by it as architecture.

This last question leaves space for free thoughts which could be useful to mention JAMstack. The answers are mostly related to suggestions about learning about JAMstack and experiences related to it.

The interview also had a question about collecting answers to whom it is beneficial and why to use JAMstack:
What criteria make you decide about the stack you are going to use?

The full answers are in the appendices.

Name	Answer
Sauli Purhonen	I prefer to keep as many strings in my hands as possible and do not mind writing more boilerplate code if it helps to keep the result more maintained and readable. For small and fastpaced projects you need to take shortcuts and use more strict environments but maybe lose some possibilities of creative solutions.
Andreas Koutsoukos	Simple to use for the developers, minimal configuration on the server-side, build in CI pipeline, just code now server-side stuff.
H. Tuohimaa	It must be fast to code, secure, and easy to understand with frontend developers.

Joe Smallwood	The client would not need to heavily maintain the solution over time. By using Netlify they do not have to run or manage any servers. They just run in the cloud and the client does not have to worry about updates or maintenance.
Anonymous	This has been Business Decision from big corporations/stakeholders to stay with the current platform (stack).

4.5 Summary

Most of the answers to the survey and the interviews contain valuable information which helps to find answers to the research questions. The respondents have 3-20 years of relevant work experience which makes the answers more accurate. 50% of the 26 survey respondents and 4 out of 5 interviewees are familiar with JAMstack so their answers contain information related to the usage and experiences of the JAMstack architecture. Those who are not familiar with the JAMstack, also contribute to the conclusion, by telling about why their current stack is satisfying or why they have not had needed to get to know JAMstack. All the information gathered through the research methods is according to the respondents' opinions and their empirical experience. All the questions of the survey, the interviews, and the answers to the interview questions have been summed up but the full and literal texts can be found in the appendices.

5 Discussion

The results of the research have been mostly affirmative towards the statements about the benefits of JAMstack found in the existing literature, which is explained and introduced in the 2nd chapter about the theory. They are confirming many characteristics of the architecture and give a mostly positive confirmation about the known facts. However, there are certain additional benefits and disadvantages of the usage in real-life projects. Through the answers to the survey and the interview, the research can introduce new sides of JAMstack and also represent the types of developers who might benefit from learning more about it, or types of projects where it can be more or less recommended to use JAMstack.

The validity of the results can be granted to the questions designed to get a better overview of the respondents. The developers who answered have 3-20 years of work experience in the software development field. Most of them are familiar with both old-school and modern web development technologies. Most of the survey respondents and all the interviewees are familiar with the tools of JAMstack if not all of them. Half of the survey respondents and 4 out of 5 interviewees are familiar with JAMstack and can compare the usage of it to their earlier experiences with different tools. They are coming from different sized enterprises and a diverse kind of software developers are represented from all around the globe and in all ages, which makes the data found more universal and less geographical location-, company size-, or years of work experience - specific.

5.1 Answers to the research questions

There is three main research question of the thesis. Some of the answers to the survey and the interviews can give answers to them additionally which cannot be found in the relevant literature. The results have been partly expected but on the other hand, they demonstrate certain unexpected views. To find out the answers the results could provide for the research question, we must go through the findings and understand what the results are, what the new information is the collected data can provide.

What are the benefits of developing a modern web application according to JAMstack which makes it stand out and simplifies web development? As jamstack.org or Biilman & Hawksworth are mentioning, a project build with the tools of JAMstack can be beneficial, mostly from the main four benefits they are discussing. Similar information can be seen from both the survey- and the interview answers. The survey answers are showing that the four main benefits are valid, with being the cheapest and most scalable as the most

significant feature. Also mentioning a faster development process. A new aspect has occurred when inspecting the survey; one benefit of JAMstack lies in its simple architecture, which is causing a shorter learning curve compared to other experiences. Most of the developers rated it as a rather easy and fast-to-learn architecture that can be considered when choosing a stack. In the interviews, the developers are also highlighting the cost-efficiency, the faster development processes, the faster rendering of the sites, and mentioning the simplicity to learn compared to earlier architectures. The compatibility with modern and popular JS frameworks also has been mentioned as an additional benefit.

What are the pros and cons of developing applications with JAMstack compared to other methodologies? The research re-assures the pros as the mentioned benefits of JAMstack both in the existing literature and both in the research findings. Additionally to its known benefits, there are other advantages to choose JAMstack for a project which are found in the answers. The companies prefer to choose a budget-friendly stack and what is easier to learn for the developer team since they would like to deliver a product on time. JAMstack has both benefits so it can be recommended to decide by using it. The answers are clear about the importance of developer experience as well, like the fact that if a technology is less complex and less back- and front-end specific, it is easier to use for more developers with different backgrounds so these are other pros why to choose JAMstack. The answers show a positive trend about recommending using it when a project which is smaller in size. In that case, JAMstack can be a beneficial solution, timewise, meaning both the learning curve for developers, both the time spent on the development processes and the web pages' load time. It is also a good choice when it comes to lowering the project's costs and easier to find a developer team who can learn it easier.

However, many disadvantages have been mentioned in the survey and the interviews. This information can be used when deciding either JAMstack is beneficial for a project or not and they are not mentioned well enough in the relevant literature. The negative aspects usually appeared for the developers after they have learned more about JAMstack or used the technologies. The main disadvantages appear in multiple answers, which are JAMstack is not suitable for projects that need a complex back-end, has larger databases, or requires constant reload due to the high amount of constant user-side input. It is due to the discussed characteristics of JAMstack, how it handles the server-side, and how it is re-building each time the content has been updated. The need for pre-build the site constantly when updated has been also mentioned as a disadvantage and that it is

time-consuming with frequent user-updates in the content. An interesting finding is a fact that it is a relatively new architecture and if there hasn't been needed to refactor a legacy project or start building a new one, there hasn't been any need to consider learning about it. If the old technologies are still the most beneficial for a company or for a product, learning a new one would consume time and money unnecessarily.

Who benefits from building an application according to JAMstack architecture? There are recommendations about the benefits of using JAMstack for many projects and developers in most existing literature, such as jamstack.org, Netlify community or Biilman, M. & Hawksworth, P.'s book. Many sources are supported by Netlify, which platform has used first the term JAMstack, that is why it is good to examine with independent research, who is it beneficial to choose to build a project according to the principals of JAMstack. Both the survey and the interviews contain valuable information about the third research problem. They agree with the existing sources in that it is beneficial for many lightweight projects where no need for a complex backend. However, the research results are highlighting the importance of project size. It is beneficial for projects with small or no database, lightweight backend, and where less frequent and less amount of user-input is needed. One answer suggests that a JAMstack project can be also a dynamic project, despite the definition of being always a static site. If the project is dynamic, then the clientside updates' amount and frequency will not affect the site negatively with constant rebuilding, otherwise, it would do so. The JAMstack is also a good choice of budget-wise for companies. That is why it is useful if the developer-team of a company is up to date to the latest technologies since the company can succeed with a smaller budget for a project. In case, if the project itself is smaller in volume and doesn't require a complex backend. JAMstack is also beneficial for developers with more front-end knowledge since it is serverless. The learning- and building-time is also shorter with JAMstack, that makes it a good choice for a developer if they have a shorter due date with the final product. From the client's point of view, JAMstack is useful because it doesn't need to maintain servers and easy to maintain compared to projects with traditional architecture.

5.2 The importance of the findings

The results of the research carried out to contain new aspects compared to the existing literature. In many points, they proved similar results but also, they contain valuable new information as well. The original term, JAMstack, is comping from Netlify, a CMS provider, which makes the listed benefits and characteristics not independent. These research results can confirm the available information's credibility and add a new perspective and valuable new information to the already existing sources. The research provides

information for deciding on which project it is worth considering using the tools of JAMstack and justifying the reasons for choosing it or rather another architecture.

5.3 Deduction

The research describes the benefits, the advantages, and disadvantages of a JAMstack project, and to whom it is the best technology to choose for a project. The research adds new information besides the existing literature and can contain information to developers and companies to make the best decision to select an architecture for the next project. The research is representing both sides of JAMstack, when is it better and when it is not the best decision to choose it. The research contains independent, non-company supported information for getting the most realistic picture of the stack.

6 Conclusion

The research’s aim was to examine the advantages and characteristics of JAMstack, a new web development architecture that does not have a large amount of literature and research about it. The thesis addresses three main research questions to find answers to through independent research. The method for collecting primary data for analysis were surveys and interviews.

The existing literature is stating four main benefits of JAMstack: cheaper scalability, better developer/client experience, better security, and better performance. The analysis of research results confirms these benefits and additional provide information related to the research problems. The additional benefits are a shorter learning curve, less need for backend development expertise, easier and cheaper maintenance for the clients. The main advantages and disadvantages of JAMstack lay in its nature. It is static so each time there is CMS-side input, the site has to re-build and it takes more time. JAMstack projects are also not suitable for back-end- heavy projects with big databases. For the same reasons, it a beneficial decision for smaller projects, or sites/applications where either the limited budget or development time suggests deciding by JAMstack. The most recommended to decide by JAMstack are especially front-end or full-stack developers with heavier frontend knowledge, companies who needs a more economical solution for smaller projects, and customers who need to lower the budget for further maintenance costs.

In conclusion, it must be repeated that the four main benefits of using JAMstack can be proven and for smaller sized projects, it is worth to decide by JAMstack. Software developers can easily master the basics of this technology and benefit from it in the long-term. The performance of different static site generators and CMS' used in a JAMstack project haven't been compared yet, or the security issues related to changeable XML tags could be a good and practical base for further research.

References

Biilman, M. & Hawksworth, P., 2020 O’Reilly Media, Modern Web Development on the JAMstack

Walliman N., 2011., Routledge. New York., Research Methods: the basics

Netlify Community, JAMstack, URL: https://jamstack.org/, accessed: 26.02.2020

Gustaffson, N. 2019 Thesis, Developing modern web applications with the React library and WordPress

Ismail, K., 13.02.2019. Read this before moving to a JAMstack-ready CMS URL: https://www.cmswire.com/web-cms/read-this-before-moving-to-a-jamstack-ready-cms/ , accessed: 26.02.2020.

Ayodeji, B., 17.05.2019, An introduction to the JAMstack, the architecture of the modern web, URL: https://www.freecodecamp.org/news/an-introduction-to-the-jamstack-the-architecture-of-the-modern-web-c4a0d128d9ca/ accessed: 26.02.2020.

Gatsby Org., URL: https://www.gatsbyjs.org/, accessed: 08.03.2020.

Abildgaard, N., R., 3.09.2018., Thesis, Perspectives on Architecture, Evolution, and Future of Web Applications, URL: https://hypesystem.dk/thesis.pdf, accessed: 08.03.2020

Dąbrowski, P., 2019., What is JAMstack, URL: https://buttercms.com/blog/what-is-jamstackm accessed: 06.03.2020.

Potok, C., 27. 08.1996, Ballantine Books, Davita's Harp

Myers, Astasia 2018. The JAMstack: It's Pretty Sweet. Memory Leak. URLhttps://medium.com/memory-leak/the-jamstack-its-pretty-sweet-e0834e4e6bb7, accessed:
13.03.2020.

Bull, M., 20. 12. 2017., JAMstack. Life in Tech. URL:
https://www.lifeintech.com/2017/12/20/jamstack/. Accessed: 13.03.2020.

Gienow, M., 26. 12. 2017., The Sweetness of JAMstack: JavaScript, APIs and Markup, URL: https://thenewstack.io/the-sweetness-of-jamstack-javascript-apis-and-markup/ accessed: 13.03.2020.

Carlise, M., 30. 11. 2018., Forget the Docker, the future is JAMstack, URL: https://hackernoon.com/forget-docker-the-future-is-jamstack-aae5bcaf4616, accessed: 14.03.2020.

Dionne, M., 16. 01. 2020., New to JAMstack? Everything You Need to Know to Get Started, URL: https://snipcart.com/blog/jamstack, accessed: 15. 03. 2020.

MDN web docs, 2020., JavaScript, URL: https://developer.mozilla.org/enUS/docs/Web/JavaScript, accessed: 23.03.2020.

Hawksworth, P., freeCodeCamp.org, 10.03.2020., JAMstack Course - Build websites that are simpler, faster, and more secure, URL: https://www.youtube.com/watch?v=A_l0qrPUJds, accessed: 23.03.2020.

Appendices

Appendix 1. The survey questions

1. What describes your profession the best?
2. How long have you been familiar with software development?
3. If you are currently working, what describes the company the most?
4. What architectures are you familiar with?
5. Have you heard of JAMstack before?
6. Have you ever used JAMstack?
7. If you have ever used JAMstack, how easy or difficult was it to learn it? 8. If you have ever used JAMstack, what benefits you have been experiencing?
9. If you have used JAMstack before, have you experienced disadvantages compared to other architectures?
10. What makes you decide which stack are you or your team are going to use for a project?
11. Are you familiar with any static site generators? (choose as many as you have been using before)
12. What JavaScript frameworks/libraries are you familiar with?
13. Are you familiar with content management systems?
14. If you are familiar with CMS, which tool do you prefer to use the most often?
15. If you have never wanted to familiarize yourself with JAMstack or never decided to use it, what were the main reasons for making that decision?
16. Here you can add anything else freely which is related to the topic. You can tell about good/bad opinions and experiences or any advice for those who are learning web development or wish to find the best tools for creating the best possible results.

Appendix 2. The interviews

Interview questions:

The questions were the same for each interviewee, which are the following:

1. How long have you been working with web development?

2. *What was your first stack when you had begun working?*

3. *What stack are you using currently?*

4. *What criteria make you decide about the stack you are going to use?*

5. *Have you been familiar with JAMstack?*

a. *If yes, what is your experience? How long did it take to learn it? Was it simpler or more difficult compared to other architecture (such as LAMP, WAMP, MEAN, etc.)? Can you explain the main advantages you have been noticing and possible disadvantages? What kinds of projects have you been using, or prefer to use it with?*

b. *If no, what made you decide not to get familiar with/ use JAMstack architecture?*

6. *Are there any more observations, experiences, or advice related to JAMstack or comparing it to other architectures you have been using before?*

7. *Anything else you would like to add?*

–

Interview 1:

Name: Sauli Purhonen

Company: Visma Consulting Oy

Work experience: 7

Answers:

How long have you been working with web development?

I 've been doing professional web development for 7 years. First six years I was doing freelance jobs for different customers with different needs, most e-commerce sites. I quit freelancing over a year ago and started as a consultant at my current company.

What was your first stack when you had begun working?

Very first stack I started and kept in use was LAMP-stack. PHP was popular when I graduated from vocational school and I ended using WordPress for many clients.

What stack are you using currently?

Nowadays I am really into Angular and Django as API-framework. Django Admin is my go-to backend to keep things clear and fast for changes.

What criteria make you decide about the stack you are going to use?

I prefer to keep as many strings in my hands as possible and don't mind writing more boilerplate code if it helps to keep the result more maintained and readable. For small
and fast-paced projects you need to take shortcuts and use more strict environments but maybe lose some possibilities of creative solutions.

Have you been familiar with JAMstack?

Yeah, I've done few projects with Netlify CMS, Strapi & Firebase. Ease of use and integration with modern frameworks as Angular, Vue & React is made simple and could be used in many projects from light to medium use cases. For larger apps, I'm still going for the dedicated backend to keep all things under control.

Interview 2:

Name: Andreas Koutsoukos
Company: Siili
Work experience: 17 years

1: About 13 years
2: LAMP
3: JAM
4: Simple to use the minimal configuration on the server-side, build in CI pipeline, just code now server-side stuff 5:
a: I have been using JAMstack kind of architecture about 5 years ago when static architecture models were hyped.
About half a week took the time to get the details about the JAMstack.
Compared to other stacks JAM is simple and good for example my kind of professionals that are designers and code.
The advantages are that I can build sites to production in less than half-hour and the cool thing is that just pushing code to GitHub repository starts the build process in Netlify. I prefer JAMstack as a replacement for WordPress kind of static sites. Also, mid-level ecommercial sites. JAMstack is not the best for sites that make interaction from users' needs like Twitter, you can make Twitter in JAMstack but hard to say is it the best stack for that kind of site.
6: Time will tell and more I am doing JAMstack projects will tell me what the real benefits are.
7: Go with the JAMstack :) It is awesome. And those fossils that are used to using it LEMP, WAMP and another stack just say them that learn a new stuff grandpa :) I know people for Angular side that are afraid of new things and trying new stuff.

Interview 3:

Name: H Tuohimaa
Company: Idean
Work experience in years: 14 Answers:

How long have you been working with web development?
Over 10 years, I started working in 2006.

What was your first stack when you had begun working?
Possible LAMP or WAMP.

What stack are you using currently?
Isomorphic JavaScript and JAMstack. Usually, in this isomorphic stack, there is Vue or React frontend and NodeJS backend. NodeJS has a frontend supportive role, for example, Proxy-gateway, backend fetch the data from back-service APIs and it collects the data, parses it to a frontend-useful format and serves it to the frontend with the cache. Back-services can be headless CMS, database, serverless function, anything with API. This isomorphic full-stack combination is usually built-in Docker containers, so publishing it will be easy in any public cloud services (AWS, Azure, Google Cloud, Heroku).

What criteria make you decide about the stack you are going to use?
It must be fast to code, secure, and easy to understand with frontend developers.

Have you been familiar with JAMstack?

a) *If yes, what is your experience? How long did it take to learn it? Was it simpler or more difficult compared to other architecture (such as LAMP, WAMP, MEAN, etc.)? Can you explain the main advantages you have been noticing and possible disadvantages? What kinds of projects have you been using, or prefer to use it with?*

b) *If no, what made you decide not to get familiar with/ use JAMstack architecture?*

Yes, typical JAMstack with static site generation is easy to understand. After all, you need some frontend code and (markdown)content. This combo is pre-built to a static site and then deployed to CDN. Really easy steps to understand. Costs are cheap because there are no virtual machine needs. Sites are fast because of pre-rendered file architecture.

When you must update a site, you can automate it with a small CI/CD pipeline.

In a big project where the client is also updating content in a back service during development, JAMstack can start to be a little bit clumsy. Example. Contentful + Netlify, if you want to update content in Contentful, you have to build Netlify every time to see what you have done, and the build time can be multiple minutes. It is not like “update a Contentful and then refresh page” -thing, it must build it first.

Are there any more observations, experiences, or advice related to JAMstack or comparing it to other architectures you have been using before?

JAMstack can work well in smaller sizes with a small team. With JAMstack you should not have 500 errors and so many backend problems in your published site because everything is a static file - it can run everywhere.

Anything else you would like to add?
Usually, these new stack architectures are much easier to understand than some old ones. They are also more straightforward to use and have a low learning curve.

Interview 4:
Name: Joe Smallwood
Company: Idean
Work experience in years: 4

Answers:

1. How long have you been working with web development?

Around 4 years "officially" as a full-time job, but I spent a few years before that experimenting with web technologies and have been coding for around 10 years

2. What was your first stack when you had begun working?

It was almost the MEAN stack. MongoDB, Express, and Node, but instead of Angular we were using a proprietary framework owned by a client built on jQuery. It tried to be similar to React but due to React licenses at the time the client built their own framework, it was horrible.

3. What stack are you using currently?

Currently using JAMstack (Gatsby, Netlify, serverless functions, Contentful, with Ecomm and CRM APIs)

4. What criteria make you decide about the stack you are going to use?

The client does not have their own tech team in house, so I was looking for a solution that they wouldn't need to heavily maintain over time. By using Netlify they do not have to run or manage any servers. Similarly, all the other "Whatever as a Service" platforms I am using (CMS, Ecommerce, etc.) they just run in the cloud and the client doesn't have to worry about updates or maintenance.

5. Have you been familiar with JAMstack?

a) If yes, what is your experience? How long did it take to learn it? Was it simpler or more difficult compared to other architecture (such as LAMP, WAMP, MEAN, etc.)? Can you explain the main advantages you have been noticing and possible disadvantages? What kinds of projects have you been using, or prefer to use it with?

So far so good, especially with Gatsby I really like the ease of setting up a website quickly that is optimized out of the box for SEO and performance. I like the mix between creating a SPA with modern tools, but in the end, just generating static HTML files for different routes.

I do not think the concept of JAMstack took a long time to learn, but Gatsby could have a steep learning curve for people unfamiliar with GraphQL or React.

Way simpler and easier to set up than other architectures I've worked with. No time needs to be spent setting up complex servers and backend services. CI is much simpler since we are just generating static files that can be uploaded to a server or static hosting service like Netlify. With a good static hosting service, JAMstack sites are incredibly fast for the end-user. Serverless functions are also easy for frontend developers to understand since they are just JavaScript.

For disadvantages, the main one is not being able to easily live preview content changes from the CMS. Since we need to build the site when content changes there's always going to be some delay before seeing the changes live. There are ways around this, but it doesn't seem trivial to set up (unless using Gatsby Cloud, but that's pretty expensive). Also, since most SSGs require us to build the entire site every time content changes JAMstack does not make sense for very dynamic websites with content changing very frequently.

7. Anything else you would like to add?

More and more SSGs using different frameworks are popping up and Netlify is doing well right now, so the future looks good for JAMstack

Interview 5:

Name: Anonymous
Work experience in years: 20+ years
Company: -

How long have you been working with web development?

I have been working in 10 years in Web Development area. I have also working experience another 10 years from Software Testing (both Functional / Integrational).

What was your first stack when you had begun working?

Well, I haven't worked directly with stacks, but my first technologies have been Perl language, MySQL, and HTML syntax with the WAMP environment.

What stack are you using currently?

Later I have used quite basic - maybe kind of old fashion nowadays - web development languages in the LAMP environment. On backend: PHP, Postgres DB and SQL procedures, NoSQL (Redis), shell scripting, and C (gnu). Yes, I had to use C for web programming too. :) On frontend: HTML, JavaScript, CSS, and some specific template languages (not open one) too.

What criteria make you decide about the stack you are going to use?

This has been Business Decision from big corporations/stakeholders to stay with the current platform (stack).

Have you been familiar with JAMstack?

a) If yes, what is your experience? How long did it take to learn it? Was it simpler or more difficult compared to other architecture (such as LAMP, WAMP, MEAN, etc.)? Can you explain the main advantages you have been noticing and possible disadvantages? What kinds of projects have you been using, or prefer to use it with?

b) If no, what made you decide not to get familiar with/ use JAMstack architecture?

Because I have mostly work with a specific platform with a specific stack. Changing the current stack would need a big business decision. We have already once made a big decision for similar Business Decision about changing platform but that was withdrawn later then as it turned just too big operational risk. :)

Are there any more observations, experiences, or advice related to JAMstack or comparing it to other architectures you have been using before?

I am not familiar with JAMstack. I have not even heard about it earlier. :)